REMARKS

Reconsideration of the above-identified application, as amended, is respectfully requested.

In the Official Action dated October 26, 2004, the Examiner first maintained the objection to the Abstract of the Disclosure as including improper legal phraseology requiring correction. In response, Applicants have amended the Abstract to remove the alleged instances of improper legal phraseology.

In the Official Action, the Examiner further indicated that the original Declaration submitted was defective. In response, applicants herein submit a new executed Declaration now believed to be in compliance with the 37 C.F.R. §1.67(a).

Further in the Office Action, the Examiner maintained the rejection of Claims 1, 5, and 6 under 35 U.S.C. §103(a) as being unpatentable over the Canal reference entitled "Very Low Power Pipeline Using Significance Compression" in view of Emma (US 4,943,908); and further maintained the rejection of Claims 2-4 and 8-9 under 35 U.S.C. §103(a) as being unpatentable over Canal in view of Emma as applied to Claim 1 above, and further in view of the Hennessy reference entitled "Computer Organization and Design: The Hardware/Software Interface". Claim 7 further stands rejected under 35 U.S.C. §103(a) as being unpatentable over Canal in view of Emma and further in view of the Brooks reference entitled "Dynamically Exploiting Narrow Width Operands to Improve Processor Power and Performance". Claims 10-18 further stand rejected under 35 U.S.C. 103(a) as being unpatentable over Canal in view of Moline (US 4,941,119).

With respect to the Examiner's rejection of Claims 1, 5, and 6 under 35 U.S.C. §103(a) as being unpatentable over Canal in view of Emma, applicants respectfully disagree in view of the amendments and remarks herein. Specifically, Claim 1 is being canceled and a new independent Claim 19 is being submitted that clarifies the invention and more clearly sets forth how the invention distinguishes over the cited prior art references.

Particularly, new Claim 19 sets forth a multistage microprocessor pipeline structure for executing instructions comprising: an instruction cache, a decoder, a register file, an arithmetic logic unit, and a cache memory. The register file, arithmetic logic unit (ALU) and cache memory organized as a plurality of slices adapted to be enabled selectively depending on a width of instruction operands and operation results, each slice comprising a reduced bit width portion of said register file, a reduced bit width portion of said arithmetic logic unit, and a reduced bit width portion of the cache memory, wherein all of said slices are enabled to operate in parallel when a full bit width processing operation is executed, or, only a minimum required number of slices are enabled to operate if an operation is determined to be narrower than full bit width, one or more said slices being enabled for operation on a cycle-by-cycle basis during the execution of instructions. In operation, instructions from the instruction cache are decoded for operation by the decoder, and registers used in the operation are detected. Contents of the register device are input to the arithmetic logic unit which executes the instruction.

According to the invention, set forth in new Claim 19, the microprocessor pipeline structure further comprises:

a register value information means for receiving from the decoder an identity of registers used in the operation, and outputting information about content values included in those registers; and,

a width determination means for receiving outputs from the decoder and the register value information means, and generating one or more width control signals used to enable one or more slices required for execution of the operation,

wherein only those slices enabled access simultaneously corresponding portions of the register file whose contents are needed for execution of the operation; the contents being input in parallel to the portions of the arithmetic logic unit which execute the operation, and the ALU results generated are written in parallel to the portions of the register file selected to receive those results.

A new Claim 20 is being added that sets forth that if an operation accesses the cache memory, only those portions of the cache memory that participate in the operation are accessed in parallel.

A new Claim 21 is being added that sets forth that a data carry operation proceeds from a lesser significant slice to a more significant slice.

Claims 2, 5-7 and 9 are being amended to change their respective indirect or direct dependencies from now canceled Claim 1 to new independent Claim 19.

In the Office Action, all claim rejections have been made primarily over the Canal reference. Even though the present invention and Canal appear to be trying to solve the same problem, there are distinct differences as to the approaches used. It is respectfully submitted that Claims 10, 18 and new independent Claim 19 are distinguishable for the following reasons:

While the approach according to the prior art focus on reducing the activity factor and hence the dynamic power consumption in the microprocessor - the methods employed in doing so are substantially different than in the present invention. The traditional microprocessor architecture, irrespective of the Instruction Set Architecture (ISA), adopts a standard basic pipeline for execution of an instruction. A typical pipeline can be executed with six basic execution steps as follows: IF->ID->RF->EX->MEM->WB, where IF is instruction fetch, ID is instruction decode, RF is register fetch, Ex is execution and effective address calculation, MEM is memory access and WB is data write back. Some of these stages could potentially be combined depending on the design's clock cycle length.

The present invention fundamentally introduces a micro-architectural approach that introduces a major stage in the traditional pipeline comprising a register value information means (register value width, sign and leading bits data), and further, re-arranges the datapath of the Register File and Execution Stage structures, to allow for slice path enablement and processing. In particular, the present invention alters the typical traditional pipeline, described above, to IF->ID->**WD**->RF->EX->MEM->**WB**, while severely modifying the WB stage. A new Width Determination (i.e., WD) means is additionally provided for determining the width of the operation of the instruction being processed. In effect, the execution of an instruction according to the new pipeline architecture proceeds as follows: 1) an instruction is fetched into the Instruction Buffer; 2) the instruction is next decoded, to determine the operation type and identify the two source registers and the destination register; 3) the source and destination registers determined IDs are used to index into the register value information means (previously referred to as RFTags) that comprises structure 210 (shown in Figure 2 of the present

specification) or alternatively, structure 510 (shown in Figure 5), and structure 710 (shown in Figure 7) and the obtained bitmask (see 720 in Figure 7), in combination with the operation type, are fed into the WD means 730 shown in Figure 7 (and alternately shown in Figure 2 as element 220 and element 520 shown in Figure 5) to determine and drive the width of the operation to be performed; 4) register read for the source registers occur here, in particular the determined register widths obtained are used to enable the slices of the register file to read; 5) the instruction is executed based on the operation width (and the execution slices) enabled; 6) if the instruction were a memory access operation, the access would occur at this stage; and, 7) the instruction result is written back into the register file after the result width value is determined while the obtained corresponding value information is written into the register value information structure. Thus, new Claim 19 sets forth the additional stages (new elements) of a microprocessor pipeline architecture comprising: a register value information means for receiving from the decoder an identity of registers used in the operation, and outputting information about content values included in those registers; and, a width determination means for receiving outputs from the decoder and the register value information means, and generates one or more width control signals used to enable one or more slices required for execution of the operation.

Clearly, Canal does not teach the concept of introducing an additional stage in the microprocessor pipeline that is used to determine the width of an instruction to be executed a-priori. Instead, Canal's approach implements use of a small number of *extension bits* appended to all data and instructions residing in the caches, registers, and functional units. It must be noted here that in the present invention, there is no necessity to have extension bits sitting in functional units. Instead, the teachings of the present

invention are such that the design includes logical wiring to enable activation of slices a-priori. When a decision is made at the WD stage about the width of the operation to be performed, the logic enables the slices concerned for the register read, execution, and write-back stages. There is no need to carry extension bits through the pipe. More importantly, the teachings of the present invention do not suggest the concept of "serialization" during execution in the pipeline.

In Canal's teachings, on the other hand, "serialization" is the fundamental technique, as Canal states in his introduction on page 181 as follows:

> Given that only significant bytes require datapath operations and storage, pipeline hardware can be simplified by using byte-serial implementations, where the datapath width may be as narrow as one byte, and **a pipeline stage is used repeatedly for the required number of significant bytes.** Although there are many alternative implementations with different degrees of parallelism, **they all have some serialization in the pipeline.** In particular, **low-order byte(s) and extension bits are first accessed and/or operated on; then additional bytes may be accessed and/or operated on if necessary.** We describe and evaluate several pipeline implementations of this type. [Emphasis Added]

It must be noted that this question of serialization about register file read and other structures is true even for the byte-parallel skewed (See Canal, Figure 7) and the byte-parallel compressed (See Canal, Figure 9) embodiments that Canal teaches.

For example, Canal's description of the byte-parallel compressed embodiment is found on page 189 as follows:

> Another alternative is a "compressed" parallel pipeline implementation (see Fig. 9). In this case, the pipeline consists of the original 5 stages. **Each instruction spends one cycle in the Ifetch stage to read 3 bytes and an additional one if a fourth byte is needed. Then it moves on to the second stage where it reads the low order byte and the extension bits. If more bytes are needed, the instruction spends one more cycle in the same stage to read all of them in parallel.** Then the instruction moves on to the ALU stage where it executes in a single cycle, using only the functional units that operate on significant bytes. **Then it moves on to the memory stage where it reads first the low order byte and the**

> **extension bits, and if needed, it spends an additional cycle to read all the remaining bytes.** If it is a store, all the significant bytes along with the extension bits are written in a single cycle. Finally, all significant bytes and the extension bits are written into the register file in a single cycle. [Emphasis Added]

Yet another major difference between the present invention's teachings and Canal's is that while the present invention stores the actual register contents (values) in the Register File without encoding them, in the Canal's case, the actual register contents (values) are encoded, with the exception of the lowest byte. In Canal's teachings, as described above, a register content is read in a "serialization" fashion, first the lowest byte plus the extension bits, and upon examining the extension bits, if more need to be read, they are then read. The full value read needs to be decoded and assembled before it can be fed into the pipe for the necessary operation. In fact, for a full width operation, multiple cycles will be spent reading, examining and decoding the values.

Respectfully, the foregoing descriptions of the fundamental differences should be clear to any one with ordinary skill in the art and the distinct differences between the present invention's teachings and Canal's teachings is readily apparent.

Even if one were to accept Canal's approach as the only way to achieve the reduction in activity factor desired in a microprocessor design for power purposes, it must be noted that the form of value compression that his teachings adopt comes with undue cost and overhead. The compressed or encoded extension bits that he adopts must be decompressed or decoded in various stages of the pipeline before they can be utilized. His scheme uses a complex encoding mechanism that may require complex logic to decode and take the necessary action. Canal's teachings do not show how any of this may be implemented or done during a microprocessor cycle. Then there is also the

question of the real cost of holding, maintaining, and transmitting extension bits from stage to stage in the pipeline.

The teachings of the present invention, on the other hand, are more practical, easy to learn, and easy to follow for implementation.

Thus, with respect to the rejection of (canceled) Claim 1 (now new Claim 19) and Claims 5 and 6 under 35 U.S.C. §103(a) as being unpatentable over Canal in view of Emma, based on explanations given above about the fundamental differences among the pipeline structures of the present invention and Canal's teaching, it is respectfully submitted that these claims should not be rejected. While the Examiner assumes that Canal's teachings have an inherent width determination mechanism, even if that assumption were correct and accepted, Canal's system would require numerous distributed width determination logic implemented all over the pipeline datapath. This definitely is different from the teaching of a centralized width determination means for an instruction as set forth in new Claim 19. Emma appears to be of no help in this regard as it is cited only for disclosure of somewhat conventional pipeline features.

Further, with respect to the rejection of Claims 2-4 and 8-9 under 35 U.S.C. §103(a) as being unpatentable over Canal in view of Emma as applied to Claim 1, and further in view of Hennessy, based on explanations given above about the fundamental differences among the pipeline structure of the present invention and Canal's teaching, Claims 2-4 and 8-9 should not be rejected and the Examiner is respectfully requested to withdraw the rejection based on 35 U.S.C. §103(a).

With specific regard to Examiner's statements in the Office Action at ¶'s12(c)-12(d), with respect to Claim 2 as being obvious in view of Hennessy, the rejection is obviated as Claim 2 has been amended to remove language directed to

examination of bits for detecting data "overlflow". Rather, Claim 2 is being amended to render it more concise and direct to the point.

However, respectfully, the concept of "overflow" as used in the present invention and set forth in Claim 8 has nothing to do with use of the sign bits as taught by Hennessy. The definition and use of overflow in the present invention has nothing to do with the traditional overflow problem. For example, if during the width determination process the values were to show that a 16-bit operation should be done, the overflow detection mechanism ensures that this operation would not overflow into the 17^{th} bit position. That is, the overflow detection mechanism of the present invention will therefore look at the leading two bits of both source operands to ensure that the result does not overflow the 16^{th} bit position. This overflow mechanism is different from what is taught by Hennessy. Furthermore, with respect to the rejection of Claim 8, it is clear that the present invention does not teach slice serialization.

With specific regard to Examiner's statements in the Office Action at ¶17 and ¶18(a) et seq., with respect to Claim 7 as being unpatentable over Canal in view of Emma as applied to Claim 1 above, and further in view of Brooks, applicants respectfully disagree. Based on explanations provided hereinabove about the fundamental differences between the pipeline structures of the present invention and Canal's teachings, it is believed that Claim 7 should not be rejected over Canal and further in view of Brooks because Brooks teaches clock gating of functional units not pipeline slices or structures in the datapath. The present invention rather teaches gating of pipeline slices or structures in the pipeline datapath which requires fundamental logic provisions unlike that taught by Brooks.

With specific regard to Examiner's statements in the Office Action at ¶¶19 and 20 of the Office Action, with respect to the rejection of Claims 10-18 as being unpatentable over Canal in view of Moline, applicants respectfully disagree based on explanations given above about the fundamental differences among the pipeline structures of the present invention and Canal's teaching. That is, with respect to the rejection of Claim 10, the conclusion is inappropriate due to the difference and the innovation exhibited by the present invention as set forth in the claim, and particularly, because it is not shown or described how Canal calculates the width of data similarly in the manner as to what is performed in the present invention. Respectfully, Moline is of no help in this regard. That is, the point raised about Moline's teachings is irrelevant to the present invention. The present invention teaches one example of performing a-priori width determination of an operation with overflow detection embedded as part of the process. Clearly what the present invention teaches is wholly different from the method Moline teaches. Moline teaches overflow prediction for a multiplication operation while the invention teaches overflow detection with respect to addition. Moline's approach finds the most significant bit (MSB) of the operands and sums them and compares them to the width of the result register. The teaching of the present invention sums up the two MSB of the slices considered and, if there is a carry, that is considered to be an overflow which is definitely a different method from that of Moline. Looking at the Examiner's statement regarding ¶20(d), respectfully, it appears the Examiner's understanding is misplaced: the method of the present invention performs overflow detection as part of the Width Determination (WD) stage of the pipeline, and this is long before the instruction enters the execution stage. WD is what enables the necessary slices for

execution. Hence, nothing from Moline's teachings motivated any of the teachings of this invention.

Based on the aforegoing reasoning, the present invention is substantially different in an implementation approach from, and functionally an improvement over Brooks et al. In particular, the approach of the present invention provides width determination and overflow detection, such that, when put together, derives a powerful solution to varying width computation that has not been shown by any prior art. The overflow detection component of the present invention allows it to achieve more robust savings in both power and performance, and compared to the Brooks et al. approach the present invention results in power reductions by about a factor of two or better. Compared to the approach proposed by Canal et al., it is believed that the approach of the invention is substantially different. Again, unlike the present approach, Canal's method does not propose a substantially new architectural implementation from the traditional microprocessor pipeline. Instead, they propose a single narrow width architecture that allows for multiple cycle execution for a wider-width computation. As much as this approach may help with some power reductions for narrow width computations, one can argue strongly that there is a potential performance bottleneck in situations with wider-width operations.

Based on the foregoing analysis, new independent Claim 19 distinguishes over the prior art by the recited limitations in the second and third paragraphs, independent claim 10 distinguishes over the prior art by the recited limitations in the third through sixth paragraphs, and independent Claim 17, distinguishes over the prior art by the limitations recited in all of its paragraphs.

This application is now believed to be in condition for allowance, and a Notice of Allowance is respectfully requested. If the Examiner believes a telephone conference might expedite prosecution of this case, it is respectfully requested that he call applicant's attorney at (516) 742-4343.

Respectfully submitted,

Steve Fischman
Registration No. 34,594

SCULLY, SCOTT, MURPHY & PRESSER
400 Garden City Plaza, Suite 300
Garden City, New York 11530
(516) 742-4343

SF:gc